



Variational Bayesian representation learning for grocery recommendation

Zaiqiao Meng¹ · Richard McCreadie¹ · Craig Macdonald¹ · Iadh Ounis¹

Received: 21 July 2020 / Accepted: 13 August 2021 / Published online: 27 August 2021
© The Author(s) 2021

Abstract

Representation learning has been widely applied in real-world recommendation systems to capture the features of both users and items. Existing grocery recommendation methods only represent each user and item by single deterministic points in a low-dimensional continuous space, which limit the expressive ability of their embeddings, resulting in recommendation performance bottlenecks. In addition, existing representation learning methods for grocery recommendation only consider the items (products) as independent entities, neglecting their other valuable side information, such as the textual descriptions and the categorical data of items. In this paper, we propose the Variational Bayesian Context-Aware Representation (VBCAR) model for grocery recommendation. VBCAR is a novel variational Bayesian model that learns distributional representations of users and items by leveraging basket context information from historical interactions. Our VBCAR model is also extendable to leverage side information by encoding contextual features into representations based on the inference encoder. We conduct extensive experiments on three real-world grocery datasets to assess the effectiveness of our model as well as the impact of different construction strategies for item side information. Our results show that our VBCAR model outperforms the current state-of-the-art grocery recommendation models while integrating item side information (especially the categorical features with the textual information of items) results in further significant performance gains. Furthermore, we demonstrate through analysis that our model is able to effectively encode similarities between product types, which we argue is the primary reason for the observed effectiveness gains.

Keywords Grocery recommendation · Representation learning · Side information · Variational Bayesian

✉ Zaiqiao Meng
Zaiqiao.Meng@glasgow.ac.uk

Richard McCreadie
Richard.McCreadie@glasgow.ac.uk

Craig Macdonald
Craig.Macdonald@glasgow.ac.uk

Iadh Ounis
Iadh.Ounis@glasgow.ac.uk

¹ School of Computing Science, University of Glasgow, Glasgow, Scotland, UK

1 Introduction

Learning latent factors (or embeddings), as an effective method for capturing and filtering features about real-world entities, has been widely used to support many tasks such as image generation (Kingma and Welling 2014), network analysis (Meng et al. 2019a) and recommendation systems (He et al. 2017). For example, in the field of recommendation systems, latent factor learning methods such as matrix factorization (Mnih and Salakhutdinov 2008; Rendle et al. 2009, 2020) and deep neural networks (He et al. 2017; Liang et al. 2018) provide competitive and often state-of-the-art performances.

Recommendation systems that use the historical customer-product interactions to provide customers with useful suggestions have been of interest to both academia and industry for many years. Grocery recommendation is an important recommendation use-case, which aims to predict which items (products) a user might choose to buy in the future based on their shopping history. Grocery recommendation systems have been widely used in many online grocery shopping platforms (e.g. Amazon) as well as in physical chain stores (e.g. Walmart) to enhance their customers' shopping experience and save their users time when choosing products. In contrast to many other recommendation use-cases, such as music or movie recommendation that represent user interactions as a sequence of user-item pairs, a user's grocery shopping history is typically represented as a sequence of *baskets* that contain multiple products (Wan et al. 2018, 2015). Hence, in this paper, we study the task of grocery recommendation, which predicts what the user will buy in the next basket, given a time-ordered sequence of that user's previously purchased shopping baskets. In this task, both the user's general interest (what items the user tends to buy) and the associations between items (what items the user tends to buy together) are important factors to account for.

Many representation learning models (Grbovic et al. 2015; Wan et al. 2017, 2018) have been proposed, which target the grocery recommendation task, among which the Skip-gram-based methods have been shown to be effective solutions and have achieved state-of-the-art performances. For example, Triple2vec (Wan et al. 2018) is an effective model that learns latent representations capturing the basket context by maximizing the likelihood of reconstructing sampled triples (e.g. $\langle \text{user1}, \text{item1}, \text{item2} \rangle$ is a triple that is sampled from user1's purchase basket with item1 and item2 co-occurring within the basket). In these models, both the user's general interest (which items the user likes) and the personalized dependencies between items (what items the user commonly includes in the same basket) are encoded by the embeddings of users and items. Furthermore, when combined with negative sampling approaches (Wan et al. 2018), these Skip-gram-based models are able to scale to very large shopping datasets. Meanwhile, through the incorporation of basket contextual information during representation learning, significant improvements in grocery recommendation have been observed (Wan et al. 2018).

However, these representation models still have several limitations: (1) they represent users and items by single deterministic points in a low-dimensional vector space, thereby limiting the expressive ability of their embeddings and recommendation performances; (2) these models are simply trained by maximizing the likelihood of recovering the purchase history, which is a point estimate solution that is prone to overfitting (Salakhutdinov and Mnih 2008) and is more sensitive to outliers when training (Barkan 2017) since it only finds a single point estimate of the parameters; (3) while grocery shopping data often has rich product side information (Chen and de Rijke 2018), the aforementioned methods assume that products are independent, neglecting the intrinsic features of products as well

as the similarities between products. Indeed, product similarities can be captured using side information, including the categorical features (e.g. product type, display aisle, etc.) and the textual information (e.g. product descriptions). Many prior works have attempted to leverage side information for other recommendation tasks (Chen and de Rijke 2018; Ning and Karypis 2012; Xiao et al. 2019), but none of them has investigated the extent to which different types of side information add value over the grocery shopping data.

To alleviate the aforementioned issues, we propose the *Variational Bayesian Context-Aware Representation* model, abbreviated as **VBCAR**,¹ which extends the existing Skip-gram-based representation models (Wan et al. 2018) to capture both the basket contextual and entity (e.g. products) side contextual information for grocery recommendation. Our VBCAR model jointly models the representation of users and items in a Bayesian manner, which represents users and items as (Gaussian) distributions and ensures that these probabilistic representations are similar to their prior distributions (using the variational auto-encoder framework Kingma and Welling 2014). In particular, the model is optimized according to the amortized inference network that learns an efficient mapping from samples to variational distributions (Shu et al. 2018), a method for efficiently approximating maximum likelihood training. Having inferred the representation vectors of users and items, we can then calculate the preference scores of items for each user based on these two types of Gaussian embeddings to make recommendations. Moreover, we extend the VBCAR model by encoding the item representation from various types of item side information, which we refer to as the VBCAR-S model, thereby providing more information about each item than is captured by the standard randomly-generated initial embedding. With the VBCAR-S model, we can encode additional item features into embeddings by extracting item features based on different types of item side information (e.g. the categorical features and the textual information). We compare our proposed models using these additional item features with several baseline methods on three real-world grocery shopping datasets. The contributions of this paper are as follows:

1. We propose a novel variational Bayesian context-aware representation model (VBCAR) for grocery recommendation that jointly learns probabilistic user and item representations while capturing the shopping basket contextual information by training on the sampled item-user-item triples.
2. We extend VBCAR for the scenario of grocery recommendation with item side information, so that the similarities between items can be captured.
3. We explore different strategies for representing textual item descriptions as well as different types of side information, to determine the most useful side information for grocery recommendation.
4. We conduct extensive experiments on three real-world grocery shopping datasets and using eight baseline methods, showing that our VBCAR model outperforms these baseline models, while integrating item side information results in further performance gains. In addition, we demonstrate that incorporating a random key with both the categorical features and the textual information can generally lead to the best performances across the used datasets.

¹ We note that initial exploratory experiments with our model have been reported at the CARS 2.0 workshop of Recsys 2019 (Meng et al. 2019b). However, the present manuscript includes many additional details of our model and further experimental results not previously presented.

The rest of this paper is organized as follows. We first discuss some related work in Sect. 2, and describe the tackled task and the Triple2vec model that our model is built on in Sect. 3. Then, we elaborate our proposed methods in Sect. 4. Afterwards we describe the experimental setting in Sect. 5 and report our experimental results in Sect. 6. Finally, we conclude the paper and highlight some future directions in Sect. 7.

2 Related work

In this section, we review the recent related work, focusing on grocery recommendation and approaches that attempt to integrate side information in the recommendation process.

2.1 Recommendation systems

Recommendation systems have attracted much interest in both academia and industry. Most of the commonly used techniques in recommendation systems can be classified into two main categories, i.e. the classical collaborative filtering methods and the neural network (NN)-based methods. Early work on recommendation systems mainly focused on modelling explicit feedback about items (i.e. rating scores) from users based on the matrix factorization (MF) algorithms (Koren 2009; Mnih and Salakhutdinov 2008; Rendle et al. 2009). An MF algorithm encodes the user-item explicit feedback as a rating matrix and predicts the rating scores of unseen items for users by completing the matrix. Many sophisticated matrix factorization techniques, such as time Singular Value Decomposition (Koren 2009), implicit factorization machines (FM) (Hu et al. 2008) and context-aware FM (Rendle et al. 2011), have been proposed to address both classical item recommendation as well as more advanced scenarios, such as time-aware (Koren 2009), implicit feedback (Hu et al. 2008) and context-aware recommendation (Rendle et al. 2011). However, these classical methods suffer from the matrix sparsity problem and cannot capture non-linear relationships between users and items. To address these issues, recent deep neural network-based recommendation methods (He et al. 2017; Liang et al. 2018; Manotumruksa et al. 2018; Sachdeva et al. 2019; Wang et al. 2019) were proposed to learn the embeddings of users and items by neural networks from user-item interactions, as well as from various types of contextual information, resulting in significant performance gains. For instance, the Neural Collaborative Filtering (He et al. 2017) model is a general framework that integrates deep neural networks into matrix factorization approaches using implicit feedback (e.g. purchase history and click behavior). Meanwhile, both the implicit and explicit relationships between items and users from both content and ratings have been used to enhance a recommendation system by learning deep latent representations using a collaborative variational auto-encoder (Li and She 2017) or graph neural networks (Liu et al. 2020). However, these approaches only focus on the general recommendation task and have no explicit objective for modeling baskets and item side information, both of which are important modeling dimensions for enhancing performance in grocery recommendation.

2.2 Grocery recommendation

Our task in this paper is to predict the items that a user will purchase next, given a sequence of shopping baskets (each containing multiple items) that they have previously bought. Some variants of the early MF-based methods can also be applied to this task, e.g. Factorizing

Personalized Markov Chains (FPMC) (Rendle et al. 2010) can model sequential behaviors between every two adjacent baskets by conducting a tensor factorization over the transition cube. Recurrent Neural Networks (RNNs) are also another category of methods that can capture global sequential features among baskets (Yu et al. 2016). However, RNNs suffer from the problem of high computational costs (Manotumruksa et al. 2018). On the other hand, skip-gram-based methods have been shown to be both effective and reasonably efficient at addressing the grocery recommendation task (Barkan and Koenigstein 2016; Grbovic et al. 2015; Wan et al. 2018). For example, the Triple2vec (Wan et al. 2018) model is both an effective and scalable solution for grocery recommendation. It first samples a large set of triples from the historical baskets, then uses the Skip-gram model to predict the occurrence probability of each triple. However, as we have discussed in the Introduction section, this method is a point estimate solution that only represents entities as deterministic embedding vectors and hence it cannot integrate the rich intrinsic features of products, leaving much room for improving the expressive ability of the embeddings. Note that none of the existing grocery recommendation models has investigated leveraging item side information and the extent to which different types of side information can add value over the more commonly used grocery shopping data.

2.3 Integrating side information within recommendation systems

With the increased availability of side information for a range of recommendation tasks, integrating such information into the recommendation systems has been widely studied (Chen and de Rijke 2018; Ning and Karypis 2012; Xiao et al. 2019). Indeed, there are variants of the MF methods, such as the hierarchical Bayesian matrix factorization (Park et al. 2013), which incorporates side information into traditional MF approaches. For example, the HIRE (Liu et al. 2019) model is a recommendation model that uses weighted matrix factorization to obtain users' and items' representations that encode both the flat and hierarchical side information, thereby improving the recommendation performance. Other works have examined the integration of side information for deep neural networks, such as the stacked denoising auto-encoder (Wang et al. 2015) and the marginalized denoising auto-encoder (Li et al. 2015). More recently, many recommendation models have explored the use of Variational auto-encoders (VAEs) (Chen and de Rijke 2018; Xiao et al. 2019; Pang et al. 2019; Wu et al. 2020) to jointly encode user ratings and side information when training, in order to overcome the (often) high-dimensionality of side information. However, most of these methods only consider a single type of item feature, for instance, only using a bag-of-words/one-hot categorical feature vector (Chen and de Rijke 2018; Wu et al. 2020) or a product category (Liu et al. 2019; Pang et al. 2019). The research scope of this paper is to build a recommendation system for the grocery shopping data, where the side information for items is typically rich of both categorical and textual information, and the user side information is commonly sparse and useless due to legitimate privacy concerns. To the best of our knowledge, our work is the first to explore different strategies for encoding textual information into item embeddings, as well as to quantify their performance impact.

3 Preliminaries

In this section, we first introduce the basic notations and the problem that we address (Sect. 3.1). Then, we briefly review a state-of-the-art representation model called Triple2vec (Wan et al. 2018) (Sect. 3.2), which is tailored to grocery recommendation, and that we build upon.

3.1 Problem definition and notations

We use $\mathcal{U} = \{u_1, u_2, \dots, u_N\}$ to denote the set of users and $\mathcal{I} = \{i_1, i_2, \dots, i_M\}$ to denote the set of items, where N is the number of users and M is the number of items. Then, in the scenario of grocery recommendation, the users' purchase history can be represented as $\mathcal{S} = \{(u, i, o) \mid u \in \mathcal{U}, i \in \mathcal{I}, o \in \mathcal{O}\}$ with $\mathcal{O} = \{o_1, o_2, \dots, o_L\}$ being the set of orders (i.e. baskets).

In this paper, we tackle the task of next-item prediction in grocery recommendation. Given $\mathcal{U}, \mathcal{I}, \mathcal{O}$ and \mathcal{S} , our aim is to infer the D -dimensional representation vectors (i.e. embeddings) of both users and items, such that these embeddings can be used to predict the next items that a user will be interested to purchase.

3.2 The Triple2vec model

Modern recommendation systems apply embedding techniques to learn latent representations of users and items from interaction data. In the grocery shopping use-case, historical purchase behaviors are represented by both individual user-product interactions and baskets representing purchase associations among products for each user. Thus, an effective embedding method for grocery recommendation should be capable of encoding product co-occurrences within a basket or across baskets from each user.

Triple2vec (Wan et al. 2018) is a recently proposed approach extended from Product2vec (Grbovic et al. 2015), which uses the Skip-gram model to capture the semantics in the users' grocery shopping basket for product representation and purchase prediction. The Skip-gram model used in Triple2vec is a neural embedding technique originally introduced in Word2vec (Mikolov et al. 2013). Specifically, given the users' purchase history $\mathcal{S} = \{\langle u, i, o \rangle \mid u \in \mathcal{U}, i \in \mathcal{I}, o \in \mathcal{O}\}$, the Triple2vec model first samples a large number of triples $\mathcal{T} = \{\langle u, i, j \rangle \mid \langle u, i, o \rangle \in \mathcal{S}, \langle u, j, o \rangle \in \mathcal{S}\}$ reflecting two items purchased by the same user in the same basket within the historical grocery shopping baskets. The model then tries to learn the latent embedding for users and items to predict the occurrence probability of these triples.

Following the basic idea of Word2vec (Mikolov et al. 2013), one can treat these sampled triples as context windows and learn latent embeddings by optimizing the log likelihood of the triple samples:

$$\begin{aligned} \log p(\mathcal{T}) &= \log \prod_{\langle i, j, u \rangle \in \mathcal{T}} p(\langle i, j, u \rangle) \\ &= \sum_{\langle i, j, u \rangle \in \mathcal{T}} (\log p(i|j, u) + \log p(j|i, u) + \log p(u|i, j)) \\ &\stackrel{\text{def}}{=} \mathcal{L}_{\text{Triple2vec}}. \end{aligned} \quad (1)$$

This leads to the training objective of Triple2vec (Wan et al. 2018). Here $p(ilj, u)$, $p(jli, u)$ and $p(uli, j)$ are the softmax formulations predicting the occurrence probability of a context entity from the embeddings of two target entities, e.g. $p(i | j, u) = \frac{\exp(\mathbf{z}_i^T(\mathbf{z}_j^i + \mathbf{z}_u^u))}{\sum_{i'} \exp(\mathbf{z}_{i'}^T(\mathbf{z}_j^i + \mathbf{z}_u^u))}$, where \mathbf{z}_u^u and $\mathbf{z}_i^i, \mathbf{z}_j^j$ are the final output embeddings of user u and items i and j , respectively. This skip-gram-based loss objective is commonly trained with the negative sampling trick that uses the Noise Contrastive Estimation (NCE) to approximate the softmax function (Mikolov et al. 2013).

4 The proposed model

The Triple2vec model only represents each user and item by single deterministic points in a low-dimensional continuous space, which limits both the expressive ability of their embeddings and the recommendation performance. To address this issue, we propose to use the Bayesian Skip-gram model (Barkan 2017) to learn the Gaussian distributions representing the users' and items' embeddings. In this section, we first present our proposed representation learning model, i.e. the Variational Bayesian Context-Aware Representation (VBCAR) model, and show how to use the learned embeddings for downstream recommendation tasks (Sect. 4.1). Then, we describe the VBCAR-S model, which extends VBCAR for integrating the item side information (Sect. 4.2).

4.1 The VBCAR model

Our VBCAR model extends Triple2vec by assuming that both the representations of users and items, i.e. \mathbf{z}^u and \mathbf{z}^i , are random variables, which are generated by the same priors. Like other probabilistic methods for embedding (Meng et al. 2019a) and recommendation systems (He et al. 2017; Liang et al. 2018), these priors are assumed to be the standard Gaussian distributions:

$$p(\mathbf{z}^u) = \mathcal{N}(0, \alpha^2 \mathbf{I}), \quad p(\mathbf{z}^i) = \mathcal{N}(0, \alpha^2 \mathbf{I}) \quad (2)$$

where α^2 is the same hyperparameter for all the priors—we use the default setting of $\alpha = 1$ in this paper, following (Kingma and Welling 2014).

However, the exact inference of the posterior density of these random variables is intractable due to the non-differentiable marginal likelihood (Kingma and Welling 2014). Variational Bayes resolves this issue by constructing a tractable lower bound of the logarithm marginal likelihood and maximizing the lower bound instead (Blei et al. 2017). Hence, we introduce a variational evidence lower bound over the observed triple samples in our task, following the Variational Autoencoding framework (Kingma and Welling 2014). To infer the users' and items' embeddings, we start by formulating the logarithm marginal likelihood of an observed triple $\mathcal{T}_i \in \mathcal{T}$:

$$\begin{aligned}
\log p(\mathcal{T}_i) &= \log \mathbb{E}_{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} \left[\frac{p(\mathcal{T}_i, \mathbf{z}^u, \mathbf{z}^i)}{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} \right] \\
&\geq \mathbb{E}_{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} \left[\log \frac{p(\mathcal{T}_i, \mathbf{z}^u, \mathbf{z}^i)}{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} \right] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} [\log p(\mathcal{T}_i | \mathbf{z}^u, \mathbf{z}^i)] \\
&\quad - KL(q_\phi(\mathbf{z}^u, \mathbf{z}^i) \| p(\mathbf{z}^u, \mathbf{z}^i))
\end{aligned} \tag{3}$$

$$\begin{aligned}
&= \mathbb{E}_{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} [\mathcal{L}_{Triple2vec}(\mathcal{T}_i)] \\
&\quad - KL(q_\phi(\mathbf{z}^u, \mathbf{z}^i) \| p(\mathbf{z}^u, \mathbf{z}^i)).
\end{aligned} \tag{4}$$

Then, the likelihood of \mathcal{T} is given by $\sum_{\mathcal{T}_i \in \mathcal{T}} \log p(\mathcal{T}_i)$, leading to the overall loss function of our VBCAR model:

$$\mathcal{L}_{VBCAR} = \sum_{\mathcal{T}_i \in \mathcal{T}} \mathbb{E}_{q_\phi(\mathbf{z}^u, \mathbf{z}^i)} [\mathcal{L}_{Triple2vec}(\mathcal{T}_i)] - KL(q_\phi(\mathbf{z}^u, \mathbf{z}^i) \| p(\mathbf{z}^u, \mathbf{z}^i)), \tag{5}$$

where $KL(\cdot \| \cdot)$ is the Kullback-Leibler (KL) divergence² and $q_\phi(\mathbf{z}^u, \mathbf{z}^i)$ is the variational distribution of the embeddings \mathbf{z}^u and \mathbf{z}^i . Following previous works (Meng et al. 2019a; Kingma and Welling 2014; Liang et al. 2018), we assume that the variational distribution can be factorized in the mean-field form of Gaussian distributions and are inferred by their respective features, i.e. $q_\phi(\mathbf{z}^u, \mathbf{z}^i) = q_{\phi_1}(\mathbf{z}^u)q_{\phi_2}(\mathbf{z}^i) = q_{\phi_1}(\mathbf{z}^u | \mathbf{F}^u)q_{\phi_2}(\mathbf{z}^i | \mathbf{F}^i)$, where \mathbf{F}^u and \mathbf{F}^i are the feature vectors of the user and item, respectively. These two variational distributions (i.e. $q_{\phi_1}(\mathbf{z}^u | \mathbf{F}^u)$ and $q_{\phi_2}(\mathbf{z}^i | \mathbf{F}^i)$) are independent and can be inferred by two different encoder networks by the features of users and items (i.e. \mathbf{F}^u and \mathbf{F}^i) as input, respectively:

$$q_{\phi_1}(\mathbf{z}^u | \mathbf{F}^u) = \mathcal{N}(\boldsymbol{\mu}^u, \boldsymbol{\sigma}^{u2} \mathbf{I}), q_{\phi_2}(\mathbf{z}^i | \mathbf{F}^i) = \mathcal{N}(\boldsymbol{\mu}^i, \boldsymbol{\sigma}^{i2} \mathbf{I}), \tag{6}$$

where $[\boldsymbol{\mu}^u, \boldsymbol{\sigma}^{u2}] = \phi_1(\mathbf{F}^u)$, $[\boldsymbol{\mu}^i, \boldsymbol{\sigma}^{i2}] = \phi_2(\mathbf{F}^i)$, ϕ_1, ϕ_2 are implemented by the full connected layers, and $\mathbf{F}^u, \mathbf{F}^i$ are the identity codes of users and items, respectively. The means of \mathbf{z}^u and \mathbf{z}^i are then used as the final embeddings of users and items respectively. Since encoding items and users using a One-hot identity representation is computationally expensive for large datasets, our VBCAR model uses randomly generated *keys* as the identity codes \mathbf{F}^u and \mathbf{F}^i for items and users respectively, where a key is a random vector initialized from the standard normal distribution. For ease of reference, we denote the key-based representations of each item and user as the *Item Key* and *User Key* representations, respectively. Figure 1a provides an overview of our VBCAR model.

4.2 Integrating item side information

Item key initialization for encoding item representations is a simple and effective solution to distinguish items, however, it cannot capture the similarities between items, which we argue is an important factor to consider. For instance, if the user previously bought a 1-litre bottle of milk and a carton of eggs ($\langle u_1, \text{milk}1L, \text{eggs}6 \rangle$), then we might want to later

² The KL divergence can be factorized into an analytic form in a similar fashion to Meng et al. (2019a).

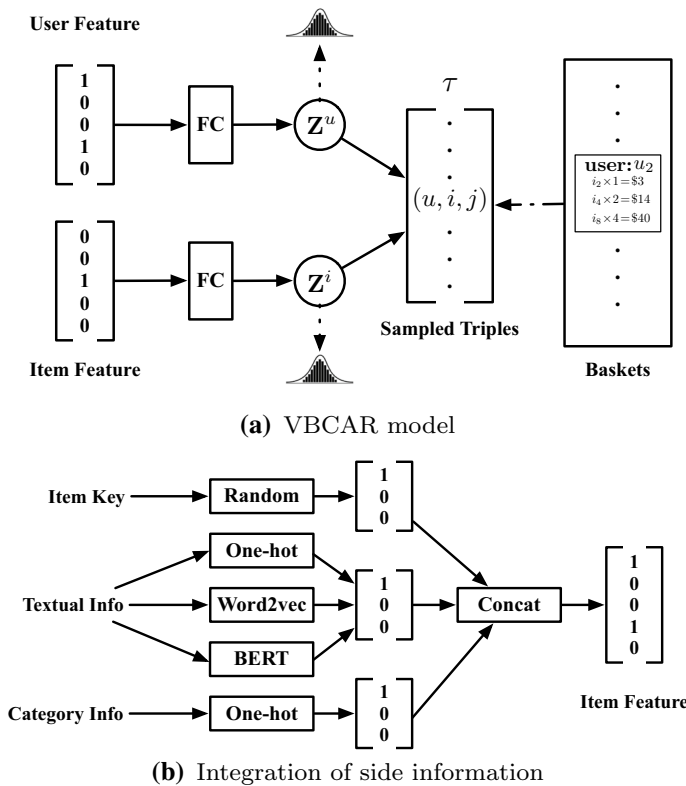


Fig. 1 Representation learning for VBCAR with item side information

recommend that they buy a 2-litre bottle of milk and a carton of eggs ($(u_1, \text{milk2L}, \text{eggs6})$), if we know that a 1-litre bottle milk (milk1L) and a 2-litre bottle of milk (milk2L) are similar. Typically, such similarity information is captured by item categories or by the (textual) item descriptions in the grocery shopping scenario. Hence, one approach to capture such similarities between items would be to encode these types of side information into the item embeddings. As discussed in Sect. 2.3, integrating the side information of items into the recommendation systems has been shown to be effective for other recommendation tasks, particularly when the available interaction data is sparse. We extend the VBCAR model into the scenario of grocery recommendation with item side information, denoted as **VBCAR-S**.

To encode item representations from item side information, we first analyzed the meta-data available about items in most of the grocery shopping scenarios/datasets. We found that there are two main types of item side information, namely, the *categorical data* such as the product's department and manufacturer, and the *textual data* such as the product's name and description. The categorical data (aka categorical features) can be easily encoded using a One-hot encoding, since the number of the categories is usually small. For the textual data or features, one can also use the One-hot encoding (i.e. bag-of-words Chen and de Rijke 2018), however, the textual data fields often have a large vocabulary size that is costly to represent using a One-hot encoder, hence we need an alternative solution. There

are many existing techniques that can encode a sequence of words into vectorial representations, such as Word2vec (Mikolov et al. 2013) and BERT (Devlin et al. 2019). However, it is not clear which technique (if any) is effective for representing similarities between products in our grocery recommendation task. Therefore, in this paper, we study the effect of different feature extraction methods over item descriptions/names to determine which representations are more useful for grocery recommendation.

Accordingly, besides the *Item Key* baseline representation of items, we also experiment with three feature extraction methods for preprocessing the item descriptions/names into vectorial representations, in addition to the encoding of categorical side information. Figure 1b shows the integration process of the side information, while the tested text encoding techniques are summarized below:

- (1) *One-hot (One)* Since encoding the full item description into One-hot category representations would be prohibitively expensive due to the large vocabulary size, we instead resort to encoding only the high frequency words (after stopwords removal) in the training triples into One-hot representations.
- (2) *Word2vec (W2v)* (Mikolov et al. 2013) Word2vec is one of the most popular methods to learn word embeddings using neural networks. We use the Google pre-trained Word2vec model³ to obtain embeddings for all the words appearing in the item descriptions. We construct the product description embedding by simply mean-pooling the word embedding vectors for all the words in the description (Yang et al. 2018).
- (3) *BERT* (Devlin et al. 2019) We also use the Bidirectional Encoder Representations from Transformers (BERT), a popular pre-trained language model for encoding words and sentences into embeddings. In particular, we use DistilBert (Wolf et al. 2019) from HuggingFace, which is a smaller and faster architecture based on BERT. The embedding of a textual description is generated via mean-pooling the BERT embeddings of the multi-words in the description. Note that BERT can be tuned to enhance performance in different tasks. However, we do not fine-tune the BERT model here, since it is prohibitively expensive to update the tens of millions of parameters in DistilBert with millions of training triples.

In our later experiments, we also combine (concatenate) different item representations together to evaluate the impact each has on the product recommendation performance. Note that integrating the side information of users can be deployed in the same way. However, the user side information is not generally available in most of the grocery shopping datasets due to legitimate privacy concerns. Hence, in this paper, we only focus on the side information of *items*.

5 Experimental setup

In this section, we describe our experimental setup, including the research questions (Sect. 5.1), datasets (Sect. 5.2), baselines (Sect. 5.3), evaluation protocol (Sect. 5.4) and parameter tuning (Sect. 5.5).

³ <http://code.google.com/archive/p/Word2vec/>.

Table 1 Statistics of the datasets

Datasets	Users	Items	Baskets	Interactions	Average size of basket	Category fields	Text fields
Instacart 25%	29,769	8045	461,030	1,429,811	3.10	2	1
Dunnhumby	2497	61,600	113,831	1,048,575	9.21	2	2
Wordline	5252	10,612	126,065	845,630	6.71	3	0

5.1 Research questions

In this paper, we evaluate our proposed model on three real-world grocery datasets, aiming to answer the following research questions:

- (RQ1) How do our proposed models perform, compared with existing methods in terms of two aspects—using item side information and without using any item side information?
- (RQ2) What is the best strategy for encoding textual information into item embeddings? To what extent do item text descriptions add value for grocery recommendation?
- (RQ3) Can our VBCAR-S model better capture item similarity than VBCAR?
- (RQ4) How do different hyperparameters (e.g., number of sampled triples, embedding dimension and hidden layer dimension) affect the performances of our models?

5.2 Datasets

We conduct experiments on three real-world grocery transaction datasets, namely *Instacart*,⁴ *Dunnhumby*⁵ and *Worldline*. The Instacart and Dunnhumby datasets are public benchmark datasets in the research community of grocery recommendation systems (Wan et al. 2018; Meng et al. 2019b), while the Worldline dataset has been collected by the Worldline company⁶ from one of its retail customers. Table 1 shows the statistics of the three datasets. Since the interaction size of the Instacart dataset is very large, it is prohibitively expensive (hours to weeks of computation time depending on the model used) to train/validate/test most recommendation algorithms (including ours) over the full dataset. Hence, following prior work (Li et al. 2017; Meng et al. 2019b), we randomly sample a subset of users and items to reduce training time. We experimented with 5%, 10%, 25% and 50% samples of the Instacart dataset, which exhibit the same trends and lead to the same conclusions, hence in this paper, we only report results on the 25% sample.⁷ The Instacart dataset contains three columns of item side information, corresponding to two types of categorical data ('aisle_id' and 'department_id') and one type of textual data ('product name'). We extracted four types of item side information from the Dunnhumby dataset, where two types (Manufacturer and Department) are regarded as categorical data and two

⁴ <http://www.instacart.com/datasets/grocery-shopping-2017>.

⁵ <http://www.dunnhumby.com/careers/engineering/sourcefiles>.

⁶ <http://www.worldline.com>.

⁷ This 25% sample is comparable in terms of the number of interactions to the other two datasets used.

types (Commodity Description and Sub Commodity Description) are textual data. The Worldline dataset only has three categorical features of items over three categorical levels. We cleaned the three datasets by removing items purchased less than 20 times as well as users who purchased more than 30 items and 10 baskets, in a similar manner to Rendle et al. (2010); He et al. (2017).

5.3 Baselines

We compare our proposed approaches to the following eight baselines:

- (1) *TopPop*: A naive method that recommends the top- k frequent items in the training set to each user in the test set.
- (2) *BPR* (Rendle et al. 2009): The Bayesian Personalized Ranking (BPR) method is a classical method for learning personalized ranking from implicit feedback.
- (3) *NeuCF* (He et al. 2017): NeuCF is a deep neural network-based collaborative filtering model that uses a multi-layer perceptron internally to learn non-linear user-item interactions.
- (4) *NGCF* (Wang et al. 2019): This is a recent recommendation framework that uses the graph neural network to integrate the user-item interactions into the embedding process.
- (5) *Triple2vec* (Wan et al. 2018): A representation model that learns the embeddings of users and items to recover product co-occurrences both within a basket and across baskets from the same user based on the Skip-gram model.
- (6) *cVAE* (Chen and de Rijke 2018): The cVAE model is a recommendation model that uses two Variational Autoencoders to simultaneously recover the user ratings and side information of items.
- (7) *SSLIM* (Ning and Karypis 2012): This is a classical sparse linear recommendation model, which learns a sparse aggregation coefficient matrix based on both the user-item purchase profiles and the item side information.
- (8) *HIRE* (Liu et al. 2019): The HIRE model is a recommendation model that uses the flat and hierarchical side information to improve the performance of recommendation. In this paper, for fair comparison, we only use its variant that eliminates the contribution of the hierarchical side information of users and items.

We note that the Triple2vec, VBCAR and VBCAR-S models are tailored for capturing the basket information in the grocery shopping data, while the cVAE, SSLIM and HIRE models can capture the item side information.

5.4 Evaluation protocol

For model evaluation, different from prior works (He et al. 2017; Manotumruksa et al. 2018), which used the leave-one-out split strategy, we split the dataset using the *temporal split strategy* (Meng et al. 2020a), where all the baskets of the datasets are split into training (80%) and test (20%) sets according to the temporal order of the baskets, and the last 20% of baskets in the training set are also used as the validation set for tuning the hyper-parameters and iterations. Note that temporal data splitting is more realistic (Quadrana et al. 2018; Campos et al. 2011) than the more commonly reported (He et al. 2017; Manotumruksa

et al. 2018) leave-one-out data splitting strategy, as evidence from the future can ‘leak’ into the learned model under the leave-one-out strategy. We treat product recommendation as an item ranking task (for each user). We report four main metrics, namely $NDCG@k$,⁸ $Precision@k$, $Recall@k$ and $MAP@k$, which are all standard ranking evaluation metrics and are widely used for evaluating recommendation systems (Valcarce et al. 2020; He et al. 2017; Wan et al. 2018).

5.5 Parameter tuning

We implemented our VBCAR and VBCAR-S models in Pytorch under the Beta-RecSys platform (Meng et al. 2020b).⁹ The baseline methods are trained using their original implementations and the hyper-parameters (e.g. the factor number of NCF He et al. 2017) are tuned based on the validation sets. To fairly compare the performances of all the models, we train/validate/test all of them using the same training/validation/testing sets, and the embedding sizes are set to 64. To reduce the time it takes to calculate the performance per user (which can be prohibitive for tens of thousands of users), we employ the standard practice of randomly sampling 100 negative items for each user in both the test sets and validation sets (Manotumruksa et al. 2017; He et al. 2017).¹⁰ For all the compared methods, the number of iterations and hyperparameters are chosen based on the validation sets, and all the reported performance results are based on the test sets. The Triple2vec, VBCAR and VBCAR-S models are trained with an RMSprop optimizer based on the same amount of sampled triples (1 million). For reference, our VBCAR and VBCAR-S models use the following hyperparameters: 512 (dimension of hidden layer in encoder), 0.001 (learning rate), and 512 (batch size).

6 Results

In this section, we describe the results of our experiments over the Dunnhumby, Instacart and Worldline datasets to answer our four research questions.

6.1 Overall recommendation performance (RQ1)

To answer RQ1, we assess the effectiveness of our VBCAR and VBCAR-S models by comparing them with eight strong recommendation methods in the literature. Table 2 shows the results of this comparison in terms of the next basket recommendation task on the Dunnhumby, Instacart and Wordline datasets. Note that for the models that use side information, i.e. cVAE, HIRE, SSLIM and VBCAR-S, the reported performances combine the item key, the Word2vec textual representation (except the Wordline dataset that does not have textual evidence) and the One-hot category representations.

Firstly, we can observe from Table 2 that both the VBCAR and VBCAR-S models show better performances than the other eight baselines under all metrics (i.e. $NDCG@10$,

⁸ Since there are no graded levels of relevance, we use {0, 1} as the relevance labels.

⁹ The source codes are available at: <https://github.com/beta-team/beta-recsys>.

¹⁰ Note that while this type of sampling will elevate the performance of all models, the comparative analysis across models is sound as the test set remains static.

Table 2 Overall performances on the three datasets ($k = 10$)

Model	Side info	Instacart 25%		
		NDCG	Recall	MAP
(a) Instacart dataset				
TopPop	✗	0.1658	0.1147	0.0737
BPR	✗	0.4071	0.1363	0.1427
NeuCF	✗	0.4646	0.2753	0.1893
NGCF	✗	0.4709	0.2805	0.2131
Triple2vec	✗	0.4678	0.2871	0.2012
VBCAR	✗	<u>0.4792</u>	<u>0.2913</u>	<u>0.2185</u>
cVAE	✓	0.4154	0.1962	0.1826
SSLIM	✓	0.4224	0.1390	0.1459
HIRE	✓	0.3220	0.1262	0.1032
VBCAR-S	✓	0.4832*	0.2937*	0.2211*
Model	Side info	Dunnhumby		
		NDCG	Recall	MAP
(b) Dunnhumby dataset				
TopPop	✗	0.1581	0.0090	0.0090
BPR	✗	0.5790	0.0728	0.0537
NeuCF	✗	0.6330	0.0710	0.0537
NGCF	✗	0.6576	0.0743	0.0521
Triple2vec	✗	0.6195	0.0874	0.0659
VBCAR	✗	<u>0.6722</u>	<u>0.0883</u>	<u>0.0741</u>
cVAE	✓	0.5705	0.0727	0.0596
SSLIM	✓	0.6035	0.0737	0.0547
HIRE	✓	0.3721	0.0625	0.0312
VBCAR-S	✓	0.6835*	0.0912*	0.0766*
Model	Side info	Worldline		
		NDCG	Recall	MAP
(c) Wordline dataset				
TopPop	✗	0.3007	0.0822	0.0505
BPR	✗	0.2995	0.0820	0.0502
NeuCF	✗	0.4271	0.1334	0.0775
NGCF	✗	0.4404	0.1425	0.0854
Triple2vec	✗	0.5697	0.1601	0.1154
VBCAR	✗	<u>0.5904</u>	<u>0.1755</u>	<u>0.1287</u>
cVAE	✓	0.3344	0.0952	0.0565
SSLIM	✓	0.3067	0.0832	0.0507
HIRE	✓	0.2995	0.0820	0.0502
VBCAR-S	✓	0.6028*	0.1809*	0.1340*

The best performing result is highlighted in bold while the second best performance is underlined; *Significant difference compared to the VBCAR model, according to the paired t -test $p < 0.05$. Note that VBCAR-S statistically outperforms all other used 8 baselines

Table 3 The performance of VBCAR-S over different types of item side information on the Dunnhumby and Instacart datasets

Types of side information					Datasets			
Key	Cat.	Textual info			Instacart 25%		Dunnhumby	
		One	W2v	BERT	NDCG@10	Recall@10	NDCG@10	Recall@10
✓	–	–	–	–	0.4792	0.2913	0.6220	0.0779
–	✓	–	–	–	0.4401	0.2682	0.6220	0.0779
–	✓	✓	–	–	0.4526	0.2803	0.6295*	0.0843*
–	✓	–	✓	–	0.4510	0.2801	0.6568*	0.0883*
–	✓	–	–	✓	0.4494	0.2762	0.6628*	0.0885*
✓	–	✓	–	–	0.4702	0.2920	0.6571*	0.0880*
✓	–	–	✓	–	0.4721	0.2901	0.6622*	0.0900*
✓	–	–	–	✓	0.4672	0.2788	0.6611*	0.0881*
✓	–	✓	✓	✓	0.4681	0.2911	0.6602*	0.0890*
✓	✓	–	–	–	0.4778	0.2856	0.6730*	0.0890*
✓	✓	✓	–	–	0.4817*	0.2992*	0.6798*	0.0897*
✓	✓	–	✓	–	0.4832*	0.2973*	0.6835*	0.0912*
✓	✓	–	–	✓	0.4801	0.2857	0.6807*	0.0897*
✓	✓	✓	✓	✓	0.4824*	0.2971*	0.6798*	0.0902*

The best performing result is highlighted in bold. *Significantly better performance compared to the model without side information, according to the paired t -test $p < 0.05$

Recall@10 and Map@10) and all datasets, which validate the value of the simple-yet-effective Variational Bayesian modelling used by VBCAR and VBCAR-S. Furthermore, our VBCAR-S model when combining the item keys with the encoded product name (and commodity descriptions) obtained the best performance over all evaluation metrics and datasets. Moreover, the performance improvements observed for VBCAR-S over the baseline methods are statistically significant (paired t -test $p < 0.05$) in all the datasets. Note that when using Precision@10, we observe the same trends as with the NDCG@10 metric, hence, we omit Precision@10 in the remainder of the evaluation. These results show that our VBCAR model outperforms the state-of-the-art recommendation models, and incorporating item side information (VBCAR-S) can further enhance the performance of the VBCAR model, which answers RQ1 and confirms observations in prior works (Liu et al. 2019; Chen and de Rijke 2018).

It is also worth noting that the TopPop baseline exhibits by far the lowest performance on both the Dunnhumby and Instacart datasets, which indicates that recommending popular products is insufficient to capture the complex purchase preferences for most of the users in many grocery shopping scenarios.

6.2 Effect over different types of item side information (RQ2)

To further understand the contribution of the different types of textual side information that we are using to the grocery recommendation performance (RQ2), we further conduct experiments over the combinations of the categorical features and the textual information encoded by three different methods (i.e. One-hot, Word2vec and BERT). Table 3 reports

the recommendation performances in the top ranks (NDCG@10 and Recall@10) over the Instacart 25% and Dunnhumby datasets.¹¹ From Table 3, which compares the VBCAR-S model when integrating both categorical features and the textual information, to the same model using only the categorical features, we can see that the addition of textual information always leads to increased performances. However, we also see from Table 3 that VBCAR-S—when using only the side information (e.g. Category, Category+One-hot and Category+Word2vec)—exhibits lower performances in terms of both the NDCG@10 and Recall@10 metrics than when using both the side information and the item key representation (across all datasets). This is expected, as these two representations provide distinct evidence. Indeed, item key representations are effective for distinguishing different items, while our side information (product names/commodity descriptions and categorical information) should in theory encode similarities between items. Hence, one reason behind the observed results may be that using only the side information to encode items makes the item representations overly similar to each other.

From Table 3, we also observe that as before, our VBCAR-S model using both the item key and side information can always outperform the model using only one of them (under both the NDCG@10 and Recall@10 metrics and over both datasets). For the Instacart dataset, the item key + One-hot encoded text, as well as the item key + Word2vec encoded text provide the best performances under the Recall@10 and NDCG@10 metrics, respectively. For the Dunnhumby dataset, the concatenated item key and Word2vec encoded text achieves the best performance under both metrics. From this, we can conclude that the text encodings add value for grocery recommendation, but that not all methods for that encoding provide equivalent performance benefits, addressing RQ2. In particular, from the results in the 6nd–13th rows of Table 3 over the two used datasets, we find that in general our VBCAR-S using categorical information can obtain better performance comparing with our VBCAR-S without using categorical information, but no consistent conclusion can be drawn as to whether using the categorical information leads to a significantly different performance. However, the performances of using both key and different side information are significantly better than that of only using the item key representation.

Furthermore, we observe that, when concatenating all the available extracted features (i.e. item key, categorical and all three text encodings), the VBCAR-S model performance does not increase (under both metrics) compared to using only one of the text encoders. Indeed, it appears that including multiple text encodings actively reduces the generalizability of the recommendation model, particularly for the Instacart dataset. Moreover, it is not the more recent (and widely seen to be effective) BERT encoding that provides the best performance, but rather the Word2vec encoding. One possible cause of this could be that we do not fine-tune the BERT model during the training. However, as already mentioned in Sect. 4.2, it is well-known that both the training and inference of the BERT model is time- and space-consuming, especially in our task setup. We leave this issue for future work due to the high computational cost. Overall, our reported results suggest that the key to effectively incorporating item side information into grocery recommendation is to properly choose a combination of different side information. For example, the combination of the random key feature and the One-hot (or Word2vec) textual feature is both an effective and computationally cheap choice on our used recommendation datasets.

¹¹ The Worldline dataset is excluded here as it does not have textual side information.

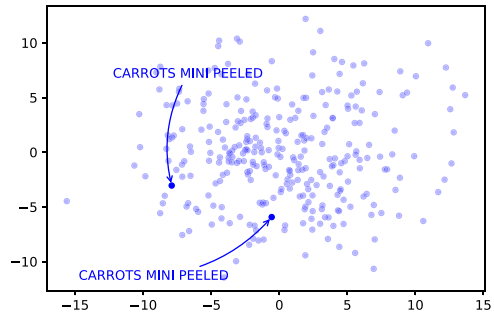
6.3 Visualization of the feature & embedding spaces (RQ3)

In order to address RQ3, we provide a case study via a 2-dimensional visualization of both the feature and embedding spaces of items and users. Specifically, we obtained two types of feature vectors (i.e. item key representations and the One-hot representations) and two types of embedding vectors (the embeddings obtained by the VBCAR model and the embeddings obtained by our VBCAR-S model with One-hot and item key features). Then we use the t-SNE tool (Maaten and Hinton 2008) to map the high-dimensional data into a 2-dimensional plane. For brevity, we only plot the top 300 frequent purchased items from the Dunnhumby dataset, which is a commonly used grocery dataset in the literature. Figure 2 shows the results of the four representation spaces. Here we choose one user (id 1545) that has an order in the test set with two purchased products (id 857503 and id 961554), where the two products have the same product name (product name: CARROTS MINI PEELED) but different sizes.

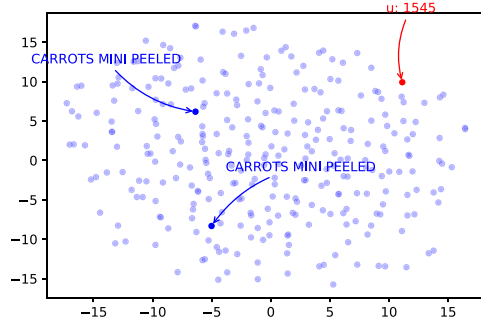
By comparing the random feature space of items (Fig. 2a) with the One-hot (Fig. 2b) feature space of items, it can be observed that the distance between the two products in the One-hot feature space is much closer than that in the random feature space, which indicates that the One-hot feature can more precisely capture the similarity of the two example products. In the embedding space of Fig. 2c, we also observe that the two products remain relatively close to each other, and that user 1545 is also encoded into a position closer to these two products. This result indicates that our VBCAR-S model is more effectively capturing item-item similarities via the additional item side information, which the original VBCAR model does not, thereby answering RQ3.

Another question that we are interested in is to investigate the impact of these similarities on the recommendation performance. Within the Dunnhumby dataset test set, we found that these two products appear within the same basket for user 1545, hence we should ideally recommend these similar items together for that user. Comparing Fig. 2b, d, we observe that our VBCAR-S model with the item key + One-hot text encoding successfully assigned high recommendation probabilities to these two products for that user since their embeddings are relatively closer to the user when compared with the item key only. In contrast, the original VBCAR model with only the item key feature did not. We can also see this graphically by comparing Fig. 2b (where the two products are separated from each other) and Fig. 2d (where the two products are close to one-another). Similar scenarios can also be found in the Instacart dataset (note that the study cannot be replicated on the Worldline dataset since the product names are not available in that dataset). These obtained results indicate that in at least some scenarios, encoding item similarities is an important factor that can enhance the grocery recommendation performance. In addition, we also noticed a few cases (less than 10%) where our model does not appropriately capture the similarity between items based on their side information. We found that there is no consistent pattern for these errors and therefore no real conclusion can be drawn. However, we suspect that these error cases are due to the heterogeneity inherent to the use of different types of side information. Recently in Meng et al. (2021), we successfully addressed this issue by integrating any heterogeneous side information using a pre-training scheme.

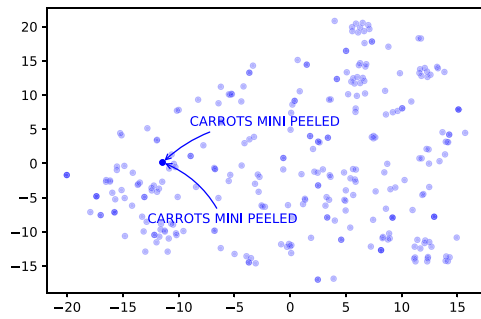
Fig. 2 2-D visualization of the feature and embedding spaces using t-SNE



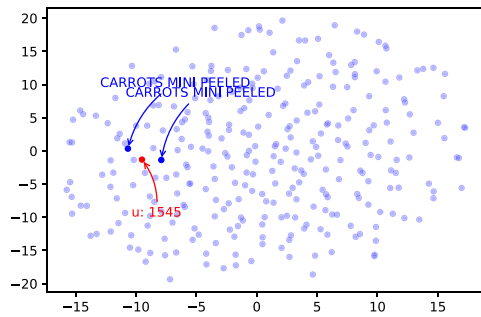
(a) Item key feature space of items



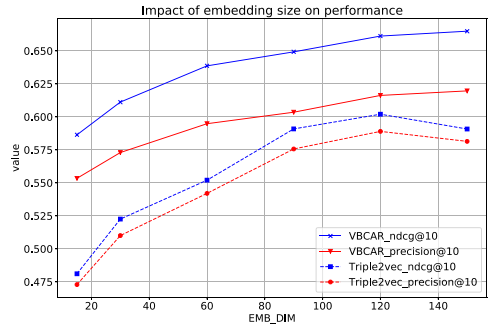
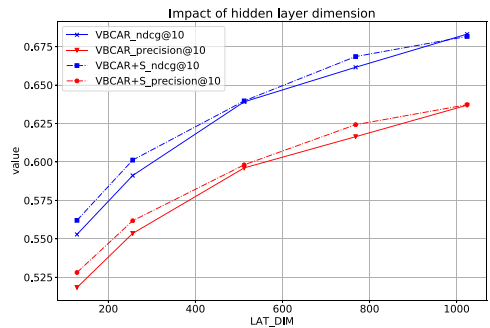
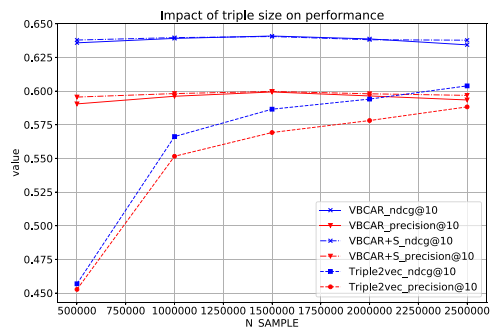
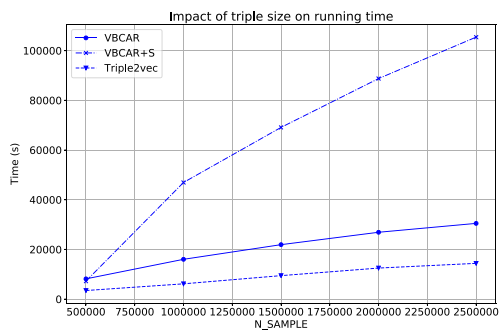
(b) Item key encoded embedding space



(c) One-hot feature space of items



(d) Item key and One-hot text encoded embedding space

Fig. 3 Analysis of hyperparameters**(a)** Embedding dimension v.s. performance**(b)** Hidden layer dimension v.s. performance**(c)** Number of triples v.s. performance**(d)** Number of triples v.s. performance

6.4 Effect of key hyperparameters (RQ4)

To answer RQ4, we analyze the influences of key hyperparameters, namely the embedding size, the hidden layer size and the number of sampled triples on both the effectiveness and efficiency of the Triple2vec, VBCAR, VBCAR-S models in the grocery recommendation task. Figure 3 shows the experimental results.

We report the performances of both the VBCAR and Triple2vec models in Fig. 3a by varying the embedding size. We can observe that increasing the embedding size substantially enhances the recommendation performance (in terms of NDCG@10 and Precision@10) for the VBCAR model and the performance peaks when the embedding size is 150, while the performance of the Triple2vec model peaks when the embedding size is 120.

Since both the VBCAR and VBCAR-S models infer embeddings of nodes and attributes by MLP with the fixed dimensional hidden layers, we analyze the performances of both the VBCAR and VBCAR-S models in Fig. 3b using different values for the hidden layer size. We can observe that increasing the hidden layer size substantially enhances the recommendation performance (in terms of NDCG@10 and Precision@10) for both the VBCAR model and the VBCAR-S model.

Figure 3c–d reports the effect of the triples number on the recommendation performance and running time. As the number of triples increases from 0.5 million to 2.5 millions, we can clearly see that the performance of Triple2vec improves, but the changes of the NDCG@10 and Precision@10 performances in both the VBCAR and VBCAR-S models are not so obvious as that in the Triple2vec model. In particular, both the VBCAR and VBCAR-S models trained with only 0.5 million sampled triples can outperform Triple2vec trained with 2.5 millions sampled triples, which means that our VBCAR model with limited sample size as input can learn more expressive representations, thereby resulting in better performance improvements compared with Triple2vec. In Fig. 3d, we can observe that the running time of all the three models increases as the number of triples increases. In addition, Triple2vec is the most efficient model under all the triple sizes while our VBCAR-S model needs more additional time to obtain the recommendations compared with the other two models. This result can be explained by the fact that VBCAR has more parameters (hidden layers) that need to be learned than Triple2vec, while VBCAR-S has some additional dimensions for encoding the item side features in comparison to the VBCAR model.

7 Conclusions

In this paper, we studied the problem of grocery recommendation with item side information. We have proposed the VBCAR model, a variational Bayesian context-aware representation model for grocery recommendation. Our model was built based on the variational Bayesian Skip-gram framework coupled with the amortized inference. To make full use of item side information, we explored the effect of different item side information on representation learning for grocery recommendation by extending the VBCAR model to leverage the item side information. In particular, we extracted both the categorical features and the textual features of items by using three representation techniques, i.e. One-hot, Word2vec and BERT, and investigated the effect of these different encoding strategies on the recommendation performance.

Our extensive experiments on three real-life datasets showed that our VBCAR and VBCAR-S models achieve better performances than eight baseline models, and that with the additional side information, VBCAR-S achieves significantly better performances than other strong baseline methods for the task of grocery recommendation. However, we found that using side information in isolation is insufficient to build an effective recommendation model, i.e. side information should only be used as supporting evidence. We also provided an illustrative case study demonstrating how the addition of side information influences the recommendation performance. Overall, our findings suggest that properly incorporating the available side information about products can significantly improve the performance of grocery recommendation systems.

As future work, we intend to examine the related task of within-basket next item prediction. We also consider that an online/living lab evaluation of recommendation systems and representation learning designed to capture item popularity over time are fertile directions for future investigation.

Acknowledgements The authors acknowledge the BigDataStack project (funded under the European Union's (Community's) Horizon 2020 research and innovation programme, Grant Agreement No. 779747).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Barkan, O. (2017). Bayesian neural word embedding. In *Proceedings of the 31st AAAI conference on artificial intelligence*, pp. 3135–3143.
- Barkan, O., & Koenigstein, N. (2016). Item2vec: Neural item embedding for collaborative filtering. In *Proceedings of the IEEE 26th international workshop on machine learning for signal processing*, pp. 1–6.
- Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518), 859–877.
- Campos, P. G., Díez, F., & Sánchez-Montañés, M. (2011). Towards a more realistic evaluation: testing the ability to predict future tastes of matrix factorization-based recommenders. In *Proceedings of the fifth ACM conference on Recommender systems*, pp. 309–312.
- Chen, Y., & de Rijke, M. (2018). A collective variational autoencoder for top-n recommendation with side information. In *Proceedings of the 3rd workshop on deep learning for recommender systems*, pp. 3–9.
- Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., & Bert. (2019). Pre-training of deep bidirectional transformers for language understanding. In *2019 Annual conference of the North American chapter of the association for computational linguistics*.
- Grbovic, M., Radosavljevic, V., Djuric, N., Bhamidipati, N., Savla, J., Bhagwan, V., & Sharp, D. (2015). E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21st ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1809–1818.
- He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182.
- Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*, pp. 263–272.
- Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes. In *Proceedings of the 2nd international conference on learning representations*, pp. 1–14.
- Koren, Y. (2009). Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 447–456.

- Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., & Ma, J. (2017). Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pp. 1419–1428.
- Li, S., Kawale, J., & Fu, Y. (2015). Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pp. 811–820.
- Li, X., & She, J. (2017). Collaborative variational autoencoder for recommender systems. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 305–314.
- Liang, D., Krishnan, R. G., Hoffman, M. D., & Jebara, T. (2018). Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, (pp. 689–698)
- Liu, S., Ounis, I., Macdonald, C., & Meng, Z. (2020). A heterogeneous graph neural model for cold-start recommendation. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 2029–2032.
- Liu, T., Wang, Z., Tang, J., Yang, S., Huang, G. Y., & Liu, Z. (2019). Recommender systems with heterogeneous side information. In *Proceedings of the world wide web conference*, pp. 3027–3033.
- Maaten, L. V. D., & Hinton, V. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov), 2579–2605.
- Manotumrukha, J., Macdonald, C., & Ounis, I. (2017). A deep recurrent collaborative filtering framework for venue recommendation. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pp. 1429–1438.
- Manotumrukha, J., Macdonald, C., & Ounis, I. (2018). A contextual attention recurrent architecture for context-aware venue recommendation. In *Proceedings of the 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 555–564.
- Meng, Z., Liang, S., Bao, H., & Zhang, X. (2019a). Co-embedding attributed networks. In *Proceedings of the 12th ACM international conference on web search and data mining*, pp. 393–401.
- Meng, Z., McCreadie, R., Macdonald, C., & Ounis, I. (2019b). Variational Bayesian context-aware representation for grocery recommendation. *Workshop on context-aware recommender systems @ RecSys*.
- Meng, Z., McCreadie, R., Macdonald, C., & Ounis, I. (2020a). Exploring data splitting strategies for the evaluation of recommendation models. In *Fourteenth ACM conference on recommender systems*, pp. 681–686.
- Meng, Z., McCreadie, R., Macdonald, C., Ounis, I., Liu, S., Wu, Y., Wang, X., Liang, S., Liang, Y., Zeng, G., et al. (2020b). Beta-rec: Build, evaluate and tune automated recommender systems. In *Fourteenth ACM conference on recommender systems*, pp. 588–590.
- Meng, Z., Liu, S., Macdonald, C., & Ounis, I. (2021). Graph neural pre-training for enhancing recommendations using side information. arXiv preprint [arXiv:2107.03936](https://arxiv.org/abs/2107.03936).
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119.
- Mnih, A., & Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *Advances in neural information processing systems*, pp. 1257–1264.
- Ning, X., & Karypis, G. (2012). Sparse linear methods with side information for top-n recommendations. In *Proceedings of the 6th ACM conference on Recommender systems*, pp. 155–162.
- Pang, B., Yang, M., & Wang, C. (2019). A novel top-n recommendation approach based on conditional variational auto-encoder. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 357–368.
- Park, S., Kim, Y.-D., & Choi, S. (2013). Hierarchical Bayesian matrix factorization with side information. In *Proceedings of the 23rd international joint conference on artificial intelligence*.
- Quadrana, M., Cremonesi, P., & Jannach, D. (2018). Sequence-aware recommender systems. *ACM Computing Surveys*, 51(4), 66.
- Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th conference on uncertainty in artificial intelligence*, pp. 452–461.
- Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on world wide web*, pp. 811–820.
- Rendle, S., Gantner, Z., Freudenthaler, C., & Schmidt-Thieme, L. (2011). Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval*, pp. 635–644.

- Rendle, S., Krichene, W., Zhang, L., & Anderson, J. (2020). Neural collaborative filtering vs. matrix factorization revisited. In *Fourteenth ACM conference on recommender systems*, pp. 240–248.
- Sachdeva, N., Manco, G., Ritacco, E., & Pudi, V. (2019). Sequential variational autoencoders for collaborative filtering. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pp. 600–608.
- Salakhutdinov, R., & Mnih, A. (2008). Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pp. 880–887.
- R. Shu, H. H. Bui, S. Zhao, M. J. Kochenderfer, and S. Ermon. Amortized inference regularization. In *Advances in Neural Information Processing Systems*, pages 4393–4402, 2018.
- Valcarce, D., Bellogín, A., Parapar, J., & Castells, P. (2020). Assessing ranking metrics in top-n recommendation. *Information Retrieval Journal*, 23, 411–448.
- Wan, M., Wang, D., Goldman, M., Taddy, M., Rao, J., Liu, J., Lymberopoulos, D., & McAuley, J. (2017). Modeling consumer preferences and price sensitivities from large-scale grocery shopping transaction logs. In *Advances in neural information processing systems*, pp. 1103–1112.
- Wan, M., Wang, D., Liu, J., Bennett, P., & McAuley, J. (2018). Representing and recommending shopping baskets with complementarity, compatibility and loyalty. In *Advances in neural information processing systems*, pp. 1133–1142.
- Wan, S., Lan, Y., Wang, P., Guo, J., Xu, J., & Cheng, X. (2015). Next basket recommendation with neural networks. *RecSys Posters*.
- Wang, H., Shi, X., & Yeung, D.-Y. (2015). Relational stacked denoising autoencoder for tag recommendation. In *Proceedings of the 29th AAAI conference on artificial intelligence*.
- Wang, X., He, X., Wang, M., Feng, F., & Chua, T.-S. (2019). Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., & Brew, J. (2019). Huggingface’s transformers: State-of-the-art natural language processing. [arXiv:1910.03771](https://arxiv.org/abs/1910.03771).
- Wu, Y., Macdonald, C., & Ounis, I. (2020). A hybrid conditional variational autoencoder model for personalised top-n recommendation. In *Proceedings of ICTIR*.
- Xiao, T., Liang, S., Shen, W., & Meng, Z. (2019). Bayesian deep collaborative matrix factorization. In *Proceedings of the 33rd AAAI conference on artificial intelligence*.
- Yang, X., Macdonald, C., & Ounis, I. (2018). Using word embeddings in twitter election classification. *Information Retrieval Journal*, 21(2–3), 183–207.
- Yu, F., Liu, Q., Wu, S., Wang, L., & Tan, T. (2016). A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th international ACM SIGIR conference on research and development in information retrieval*, pp. 729–732.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.